**DAY 2 PROGRAMS**

1. Generate a program for Pascal triangle.

Estimate the time complexity for the row=5

Code:

#include<stdio.h>

int main()

{

int rows, coef = 1, space, i, j;

int count=0;

printf("Enter the number of rows: ");

scanf("%d", &rows);

for (i = 0; i<rows; i++)

{

count++;

for (space = 1; space <= rows - i; space++)

printf(" ");

count++;

for (j = 0; j <= i; j++)

{

count++;

if(j == 0 || i == 0){

coef = 1;

count++;

}

else

{

coef = coef \* (i - j + 1) / j;

}

count++;

printf("%4d", coef);

}

printf("\n");

count++;

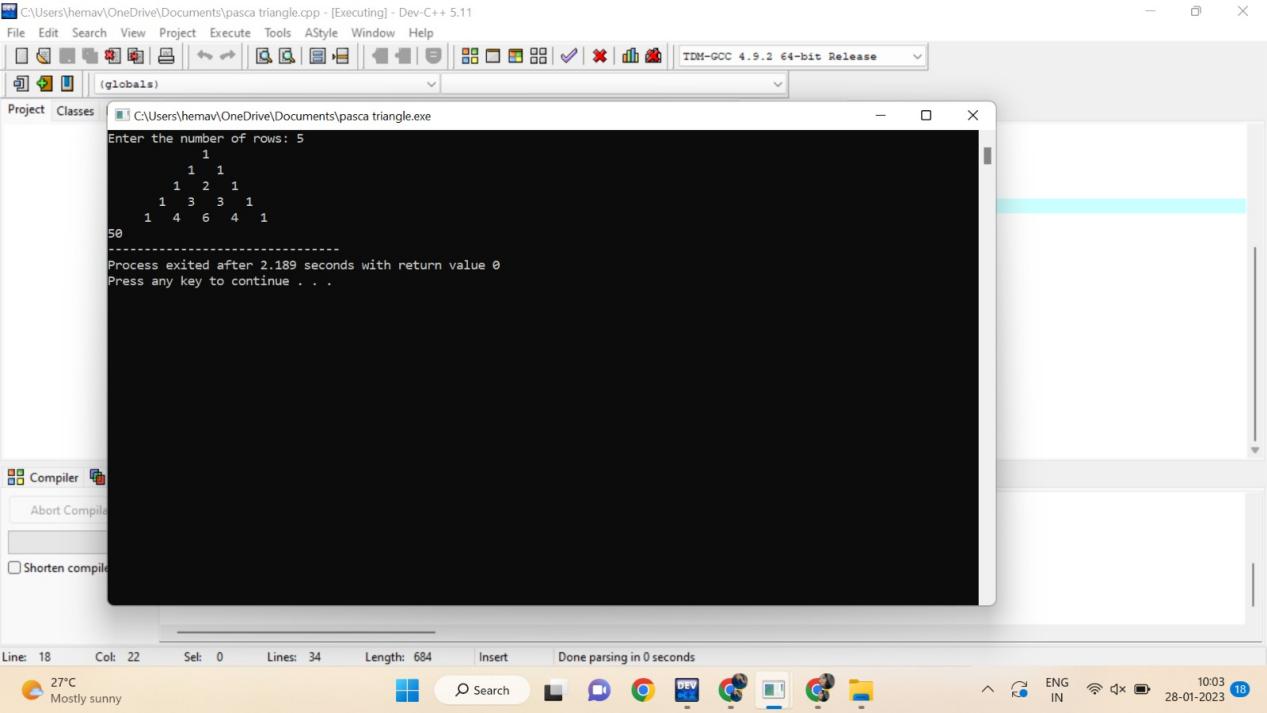
}

printf("%d",count);

return 0;

}

OUTPUT:



1. Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.integer target. Write a program to search a number in a list using binary search and estimate time complexity

CODE:

#include <stdio.h>

int main()

{

int i, low, high, mid, n, key, array[100];

int count=0;

printf("Enter number of elements");

scanf("%d",&n);

printf("Enter %d integers", n);

for(i = 0; i < n; i++)

scanf("%d",&array[i]);

printf("Enter value to find");

scanf("%d", &key);

low = 0;

count++;

high = n - 1;

count++;

mid = (low+high)/2;

count++;

while (low <= high) {

count++;

if(array[mid] < key)

low = mid + 1;

else if (array[mid] == key) {

count++;

printf("%d found at location %d ", key, mid+1);

break;

}

else

high = mid - 1;

mid = (low + high)/2;

count++;

}

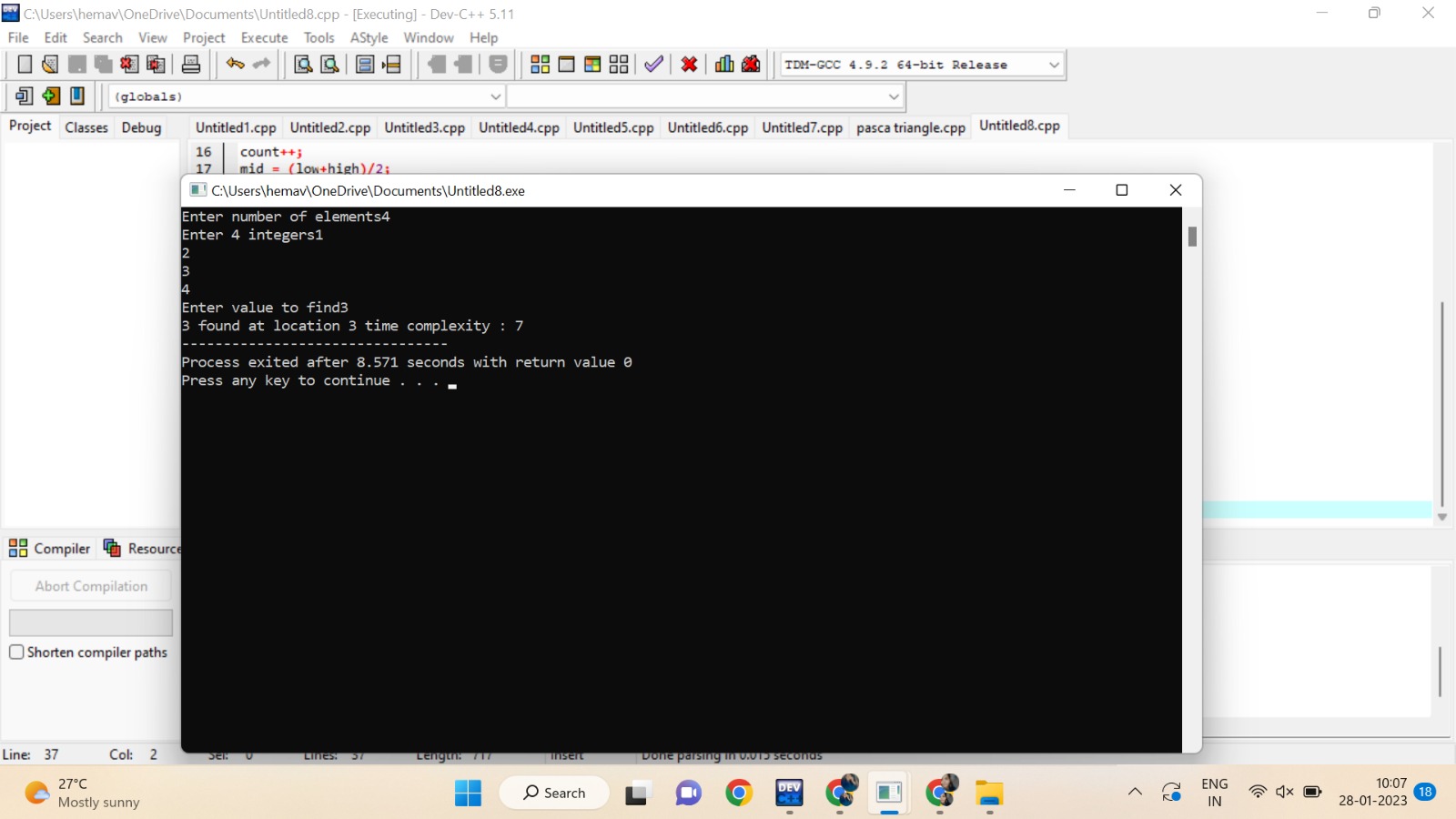
if(low > high)

printf("Not found! %d isn't present in the list.n", key);

printf("time complexity : %d",count);

return 0;

}

OUTPUT:

3. Write a program to check the given is Armstrong or not.

The k-digit number N is an Armstrong number if and only if the k-th power of

each digit sums to N.

Given a positive integer N, return true if and only if it is an Armstrong number.

Input : 153 Input : 419

Output : True Output : False

Also find the time complexity.

CODE:

#include<stdio.h>

int main()

{

int n,r,sum=0,temp;

int count=0;

printf("enter the number=");

scanf("%d",&n);

temp=n;

while(n>0)

{

count++;

r=n%10;

count++;

sum=sum+(r\*r\*r);

count++;

n=n/10;

count++;

}

count++;

if(temp==sum)

printf("true ");

else

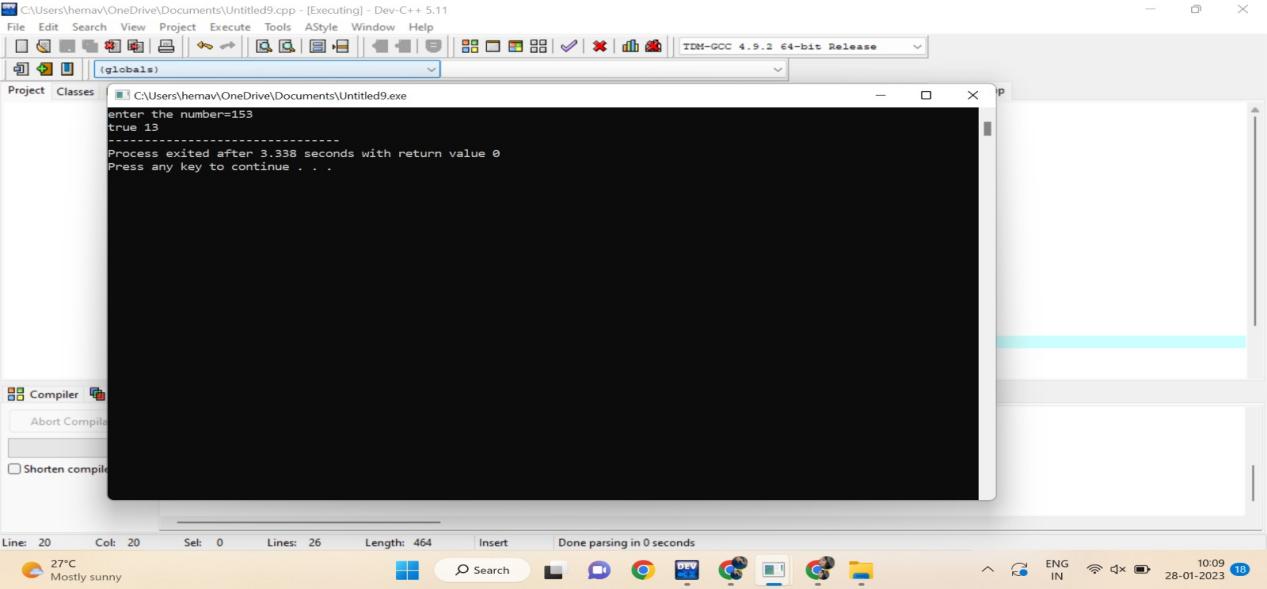
printf("false");

printf("%d",count);

return 0;

}

OUTPUT:



4.Write a C program to perform Strassen’s Matrix Multiplication for the 2\*2 matrix elements.

And Estimate time complexity.

A Matrix= (3, 5,-4, 7) B Matrix – (9,-2, 4, 8)

CODE:

#include<stdio.h>

int main()

{

int a[2][2], b[2][2], c[2][2], i, j;

int m1, m2, m3, m4 , m5, m6, m7;

int count=0;

printf("Enter the 4 elements of first matrix: ");

count++;

for(i = 0;i < 2; i++)

{

count++;

for(j = 0;j < 2; j++){

count++;

scanf("%d", &a[i][j]);

}

}

count++;

count++;

printf("Enter the 4 elements of second matrix: ");

for(i = 0; i < 2; i++){

count++;

for(j = 0;j < 2; j++){

count++;

scanf("%d", &b[i][j]);

}

}

count++;

printf("\nThe first matrix is\n");

for(i = 0; i < 2; i++){

count++;

printf("\n");

for(j = 0; j < 2; j++){

count++;

printf("%d\t", a[i][j]);

}

}

count++;

count++;

printf("\nThe second matrix is\n");

for(i = 0;i < 2; i++){

count++;

printf("\n");

for(j = 0;j < 2; j++){

count++;

printf("%d\t", b[i][j]);

}

}

count++;

count++;

m1= (a[0][0] + a[1][1]) \* (b[0][0] + b[1][1]);

count++;

m2= (a[1][0] + a[1][1]) \* b[0][0];

count++;

m3= a[0][0] \* (b[0][1] - b[1][1]);

count++;

m4= a[1][1] \* (b[1][0] - b[0][0]);

count++;

m5= (a[0][0] + a[0][1]) \* b[1][1];

count++;

m6= (a[1][0] - a[0][0]) \* (b[0][0]+b[0][1]);

count++;

m7= (a[0][1] - a[1][1]) \* (b[1][0]+b[1][1]);

count++;

c[0][0] = m1 + m4- m5 + m7;

count++;

c[0][1] = m3 + m5;

count++;

c[1][0] = m2 + m4;

count++;

c[1][1] = m1 - m2 + m3 + m6;

count++;

printf("\nAfter multiplication using Strassen's algorithm \n");

for(i = 0; i < 2 ; i++){

count++;

printf("\n");

for(j = 0;j < 2; j++){

count++;

printf("%d\t", c[i][j]);

}

}

count++;

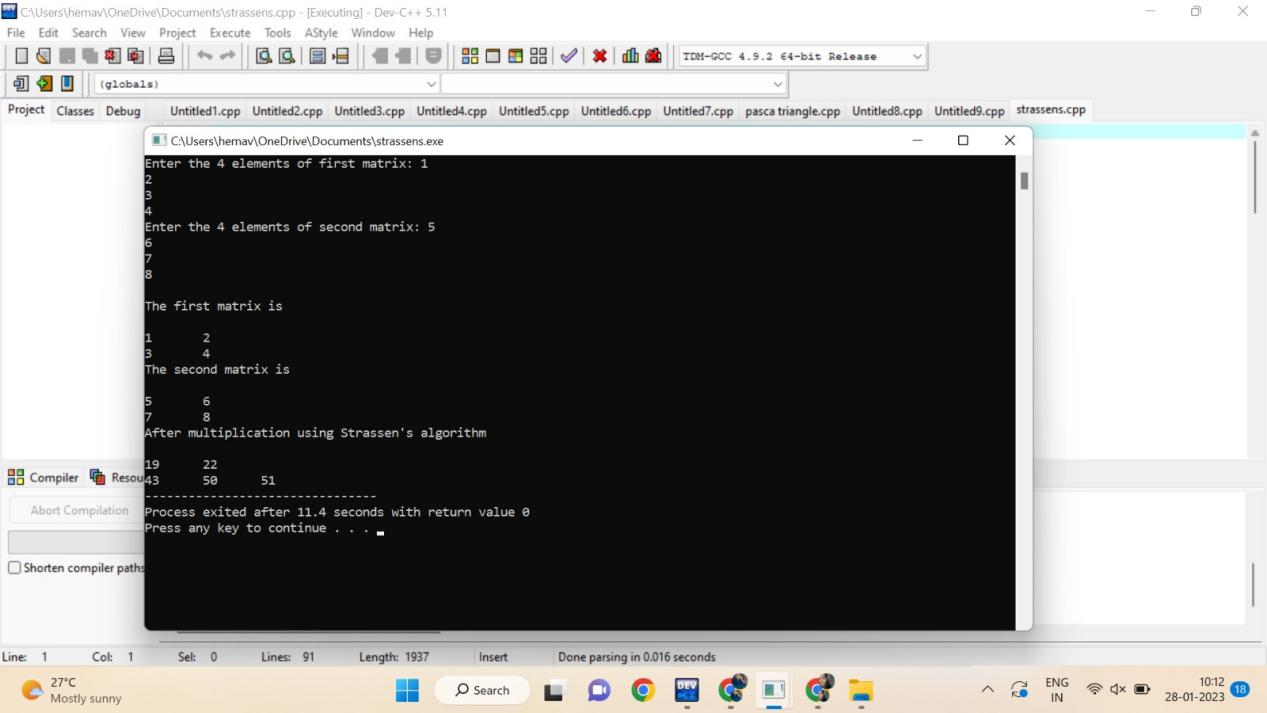
count++;

printf("%d",count);

return 0;

}

OUTPUT:



5. Compute the program to find the GCD of two numbers. And also find the time complexity.

Perform the test cases for the given set of no’s

(36,48) B. (144, 90) C. (-56,88) D. (84,84)

CODE:

#include <stdio.h>

int main()

{

int n1, n2, i, GCD\_Num;

int count=0;

printf ( " Enter any two numbers: \n ");

scanf ( "%d %d", &n1, &n2);

for( i = 1; i <= n1 && i <= n2; ++i)

{

count++;

if (n1 % i ==0 && n2 % i == 0)

GCD\_Num = i;

count++;

}

count++;

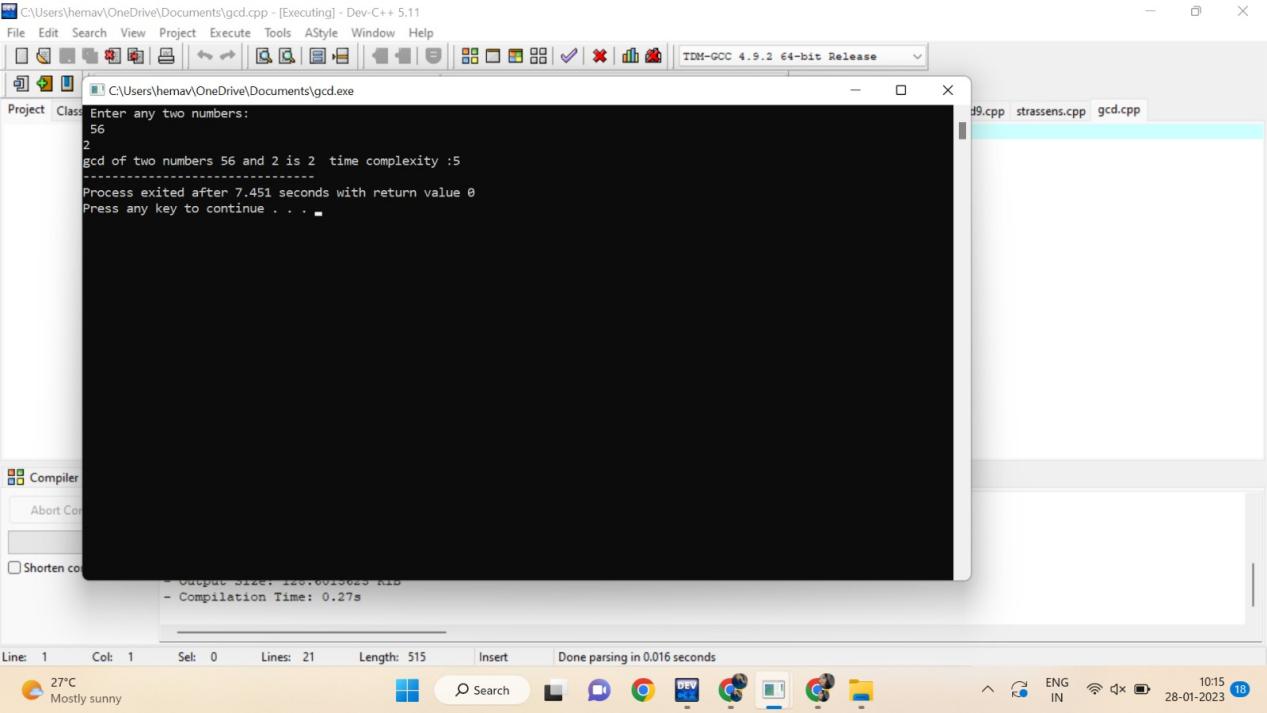
printf ("gcd of two numbers %d and %d is %d ", n1, n2, GCD\_Num);

printf("time complexity :%d ",count);

return 0;

}

OUTPUT:



6. Compute Binomial coefficient for n=8, k=8 using dynamic programming

Using condition such as

I nCk =1 if k=0 or n=k

II nCk – (n-1)Ck-1 + (n-1)Ck for n>k>0

CODE:

#include <stdio.h>

int count=0;

int bin\_table(int val) {

for (int i = 0; i <= val; i++) {

count++;

printf("%2d", i);

int num = 1;

for (int j = 0; j <= i; j++) {

count++;

if (i != 0 && j != 0)

num = num \* (i - j + 1) / j;

count++;

printf("%4d", num);

}

printf("\n");

count++;

}

}

int main() {

int value = 5;

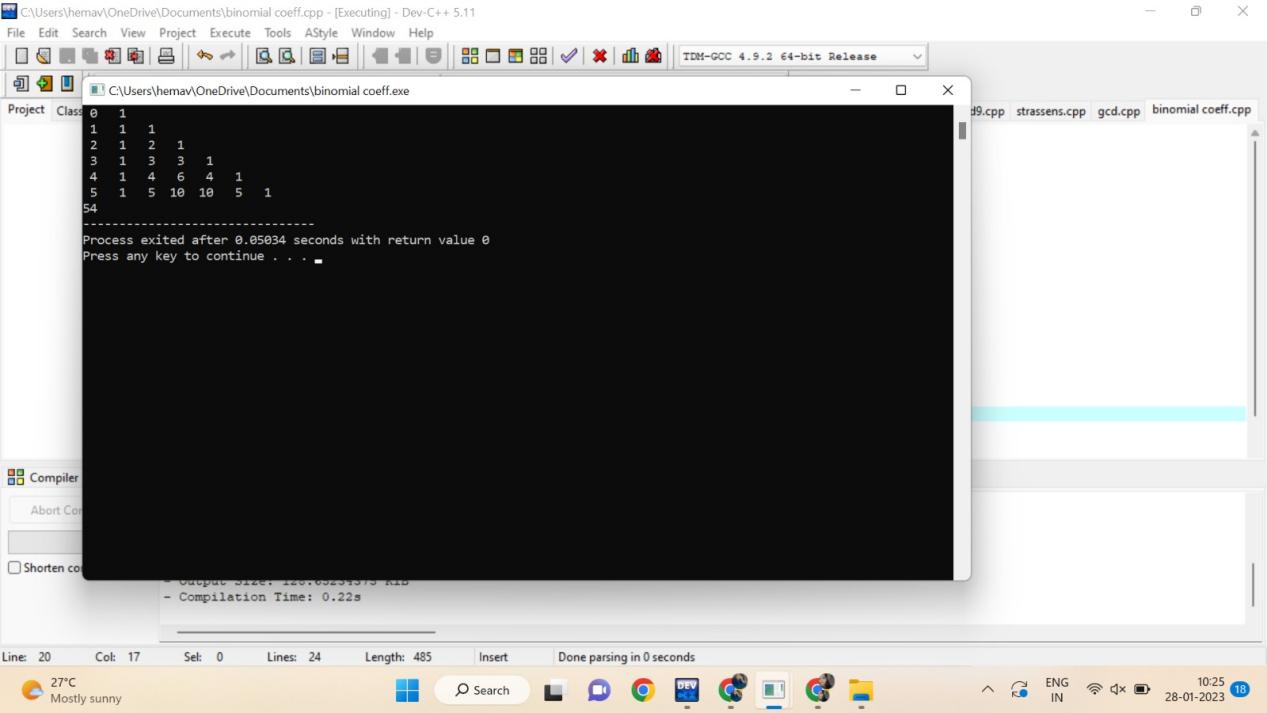
bin\_table(value);

printf("%d",count);

return 0;

}

OUTPUT:



7 Write a program for to perform liner search and estimate time complexity. Compute the

amount of time for completion.

Input series

A = (56,89,7,13,75, 23, 8, 12) Key element 75

B= (89,45 -23,45,0, 44, 2) Key element 0

C= (45,67,56,A,34,-2,100) Key element 90

CODE:

#include<stdio.h>

int main()

{

int array[100], search, c, n;

int count=0;

printf("Enter number of elements in array\n");

scanf("%d", &n);

printf("Enter %d integer(s)\n", n);

for (c = 0; c < n; c++)

{

count++;

scanf("%d", &array[c]);

}

count++;

printf("Enter a number to search\n");

scanf("%d", &search);

for (c = 0; c < n; c++)

{

count++;

if (array[c] == search)

{

printf("%d is present at location %d.\n", search, c+1);

break;

}

}

count++;

if (c == n)

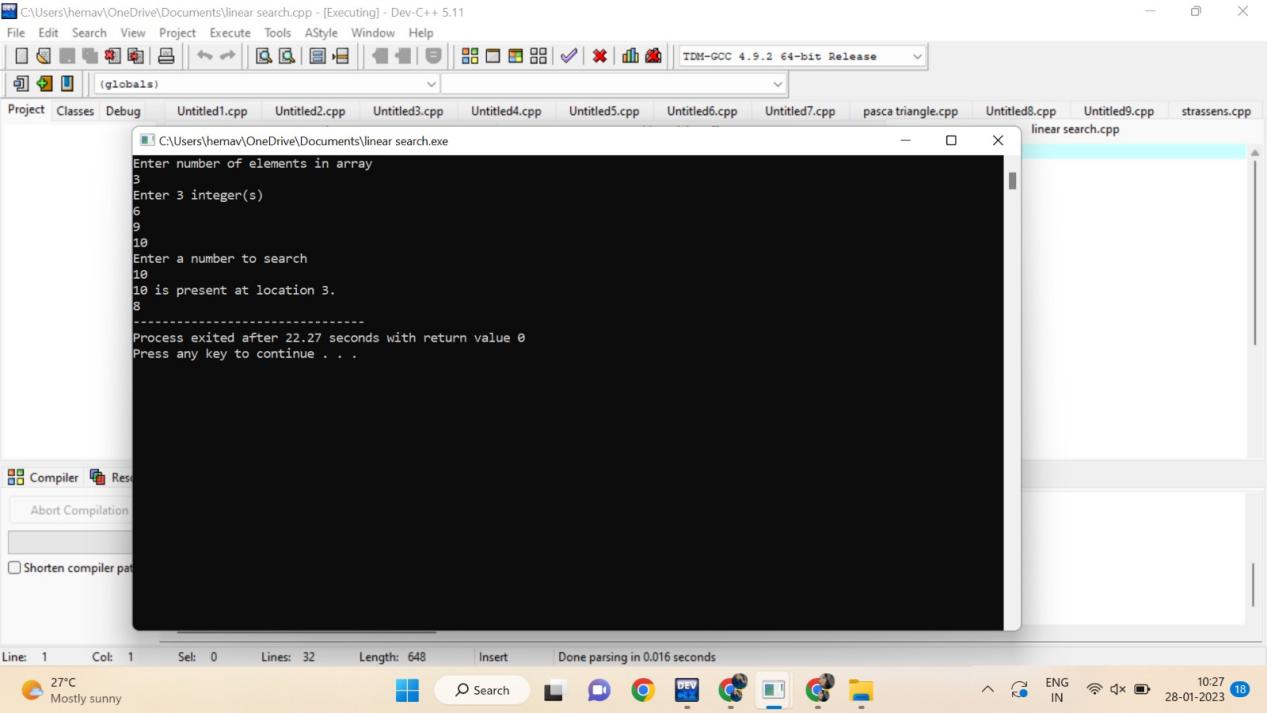
printf("%d isn't present in the array.\n", search);

printf("%d",count);

return 0;

}

OUTPUT:



8. Write a program to find the largest element value in an array. Estimate the time complexity and no of

comparison for the given set of values.

CODE:

#include <stdio.h>

int main() {

int n;

int count=0;

double arr[100];

printf("Enter the number of elements (1 to 100): ");

scanf("%d", &n);

count++;

for (int i = 0; i < n; ++i) {

count++;

printf("Enter number%d: ", i + 1);

scanf("%lf", &arr[i]);

}

for (int i = 1; i < n; ++i) {

count++;

if (arr[0] < arr[i]) {

arr[0] = arr[i];

}

count++;

}

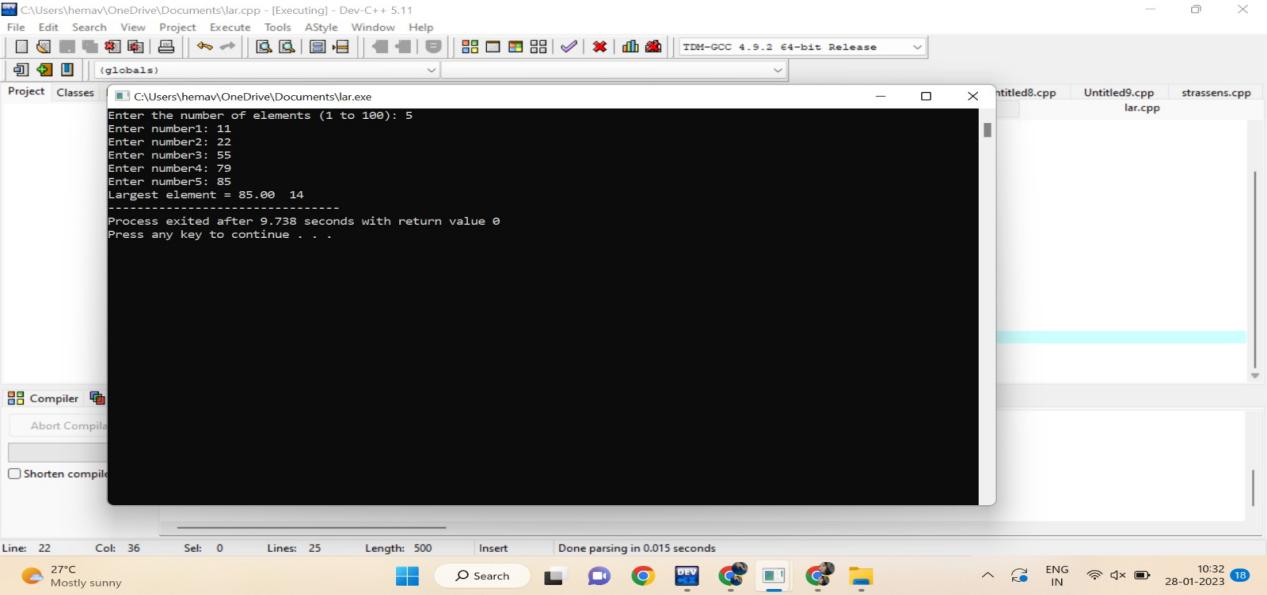
printf("Largest element = %.2lf ", arr[0]);

printf("%d",count);

return 0;

}

OUTPUT:



9. Write a program to find the factorial (fact)of a number and to estimate time complexity.

Condition such as i. n=0, return 1 otherwise fact (n-1) \* n

CODE:

#include <stdio.h>

int main() {

int n, i;

int count=0;

unsigned long long fact = 1;

printf("Enter an integer: ");

scanf("%d", &n);

count++;

if (n < 0)

printf("Error! Factorial of a negative number doesn't exist.");

else {

for (i = 1; i <= n; ++i) {

fact \*= i;

count++;

}

printf("Factorial of %d = %llu ", n, fact);

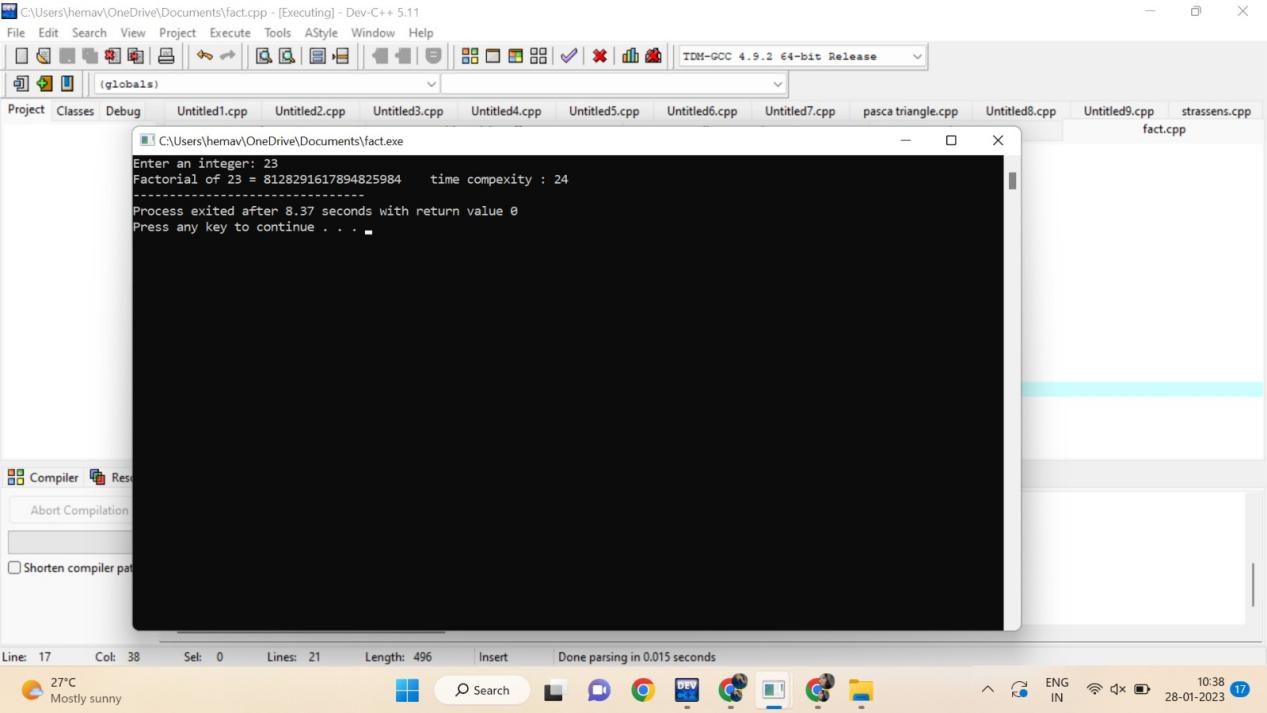
printf(" time compexity : %d ",count);

}

return 0;

}

OUTPUT:



10. [Program to Find Even Sum of Fibonacci Series Till number N](https://www.geeksforgeeks.org/java-program-to-find-sum-of-fibonacci-series-numbers-of-first-n-even-indexes/)?(day 2)

Sample Input: n = 4

Sample Output: 33

(N = 4, So here the Fibonacci series will be produced from 0th term till 8th term: 0, 1, 1, 2, 3, 5, 8, 13, 21

Sum of numbers at even indexes = 0 + 1 + 3 + 8 + 21 = 33). Find the time complexity.

CODE:

11. Write a program to print the first n perfect numbers. (Hint Perfect number means **a positive** integer that is equal to the sum of its proper divisors)

Sample Input:

N = 3

Sample Output:

First 3 perfect numbers are: 6 , 28 , 496

Test Cases:

1. N = 0
2. N = 5
3. N = -2
4. N = -5
5. N = 0.2

CODE:

#include<stdio.h>

int fab(int n){

int n1=0,n2=1,n3,c=0,sum=0,ini=0;

int count=0;

printf("%d %d ",n1,n2);

while(c<n){

count++;

n3=n1+n2;

count++;

printf("%d ",n3);

if(ini%2==0){

count++;

sum+=n3;

count++;

c++;

count++;

}

ini++;

count++;

n1=n2;

count++;

n2=n3;

count++;

}count++;

printf("\nsum: %d\n",sum);

printf("count: %d",count);

return 0;

}

void main(){

int n;

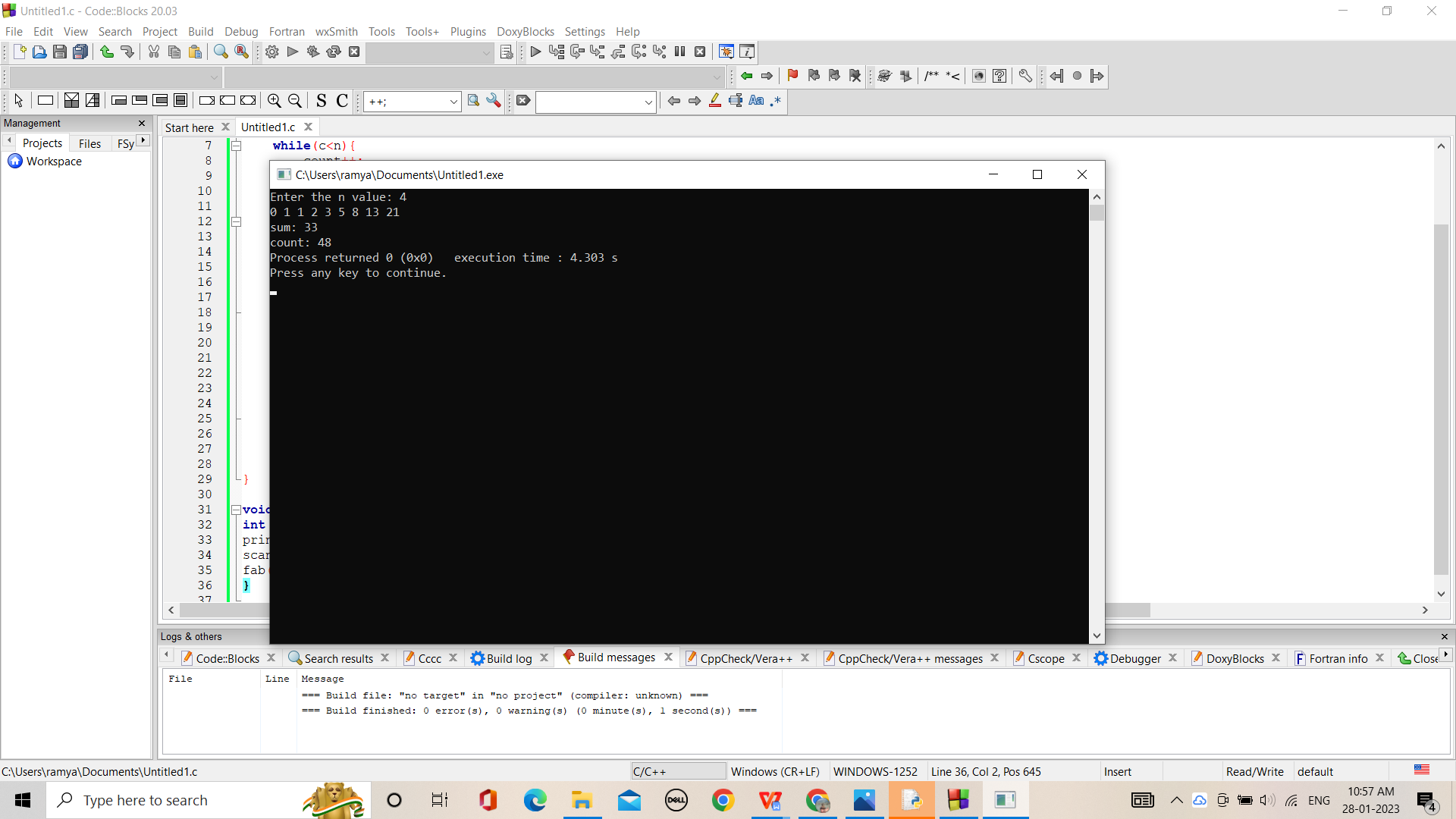
printf("Enter the n value: ");

scanf("%d",&n);

fab(n);

}

OUTPUT:



11. Write a program to print the first n perfect numbers. (Hint Perfect number means **a positive** integer that is equal to the sum of its proper divisors)

Sample Input:

N = 3

Sample Output:

First 3 perfect numbers are: 6 , 28 , 496

Test Cases:

1. N = 0
2. N = 5
3. N = -2
4. N = -5
5. N = 0.2

Find the time complexity

CODE:

#include <stdio.h>

#include<math.h>

int count=0;

int isPerfect(long long int n) {

long long int dsum = 0;

long long int i;

count++;

for (i = 1; i <= sqrt(n); ++i) {

count++;

if (n % i == 0) {

count++;

if (i == n / i) {

dsum += i;

}

else {

dsum += i;

dsum += n / i;

count++;

}

count++;

}

count++;

}

count++;

dsum = dsum - n;

count++;

if (dsum == n) return 1;

else return 0;

}

int isPrime(long long int n) {

if (n == 1)

return 0;

for (int i = 2; i <= sqrt(n); ++i) {

count++;

if (n % i == 0)

return 0;

}

return 1;

count++;

}

int main() {

long long int n, i, temp;

printf("Enter n: ");

scanf("%d", &n);

count++;

i = 1;

while (n > 0) {

count++;

if (isPrime(i) == 1) {

temp = pow(2, i - 1) \* (pow(2, i) - 1);

count++;

if (isPerfect(temp) == 1) {

printf("%d ", temp);

n = n - 1;

count++;

}

}

i = i + 1;

count++;

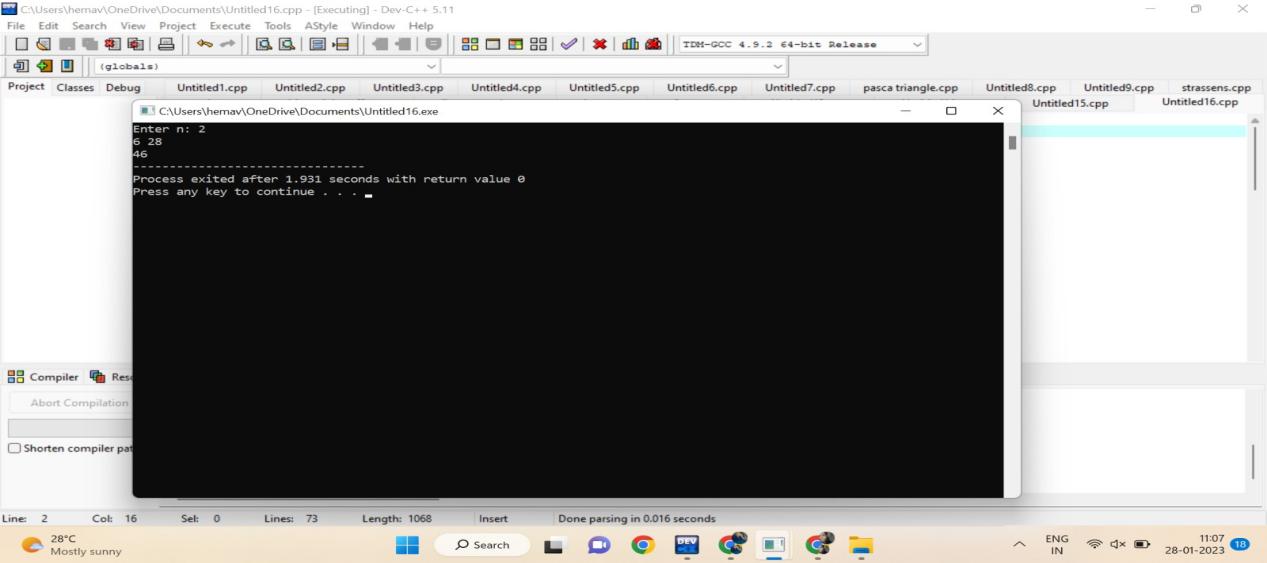
}

printf("\n");

printf("%d",count);

}

OUTPUT:



12. Write a program using choice to check

Case 1: Given string is palindrome or not

Case 2: Given number is palindrome or not

Sample Input:

Case = 1

String = MADAM

Sample Output:

Palindrome

Test cases:

1. MONEY
2. 5678765
3. MALAY12321ALAM
4. MALAYALAM
5. 1234.4321

CODE:

#include <stdio.h>

#include <string.h>

int main(){

char string1[20];

int i, length;

int flag = 0;

int count=0;

printf("Enter a string:");

scanf("%s", string1);

length = strlen(string1);

count++;

for(i=0;i < length ;i++){

count++;

if(string1[i] != string1[length-i-1]){

flag = 1;

break;

count++;

}

count++;

}

count++;

if (flag) {

printf("%s is not a palindrome ", string1);

}

else {

printf("%s is a palindrome ", string1);

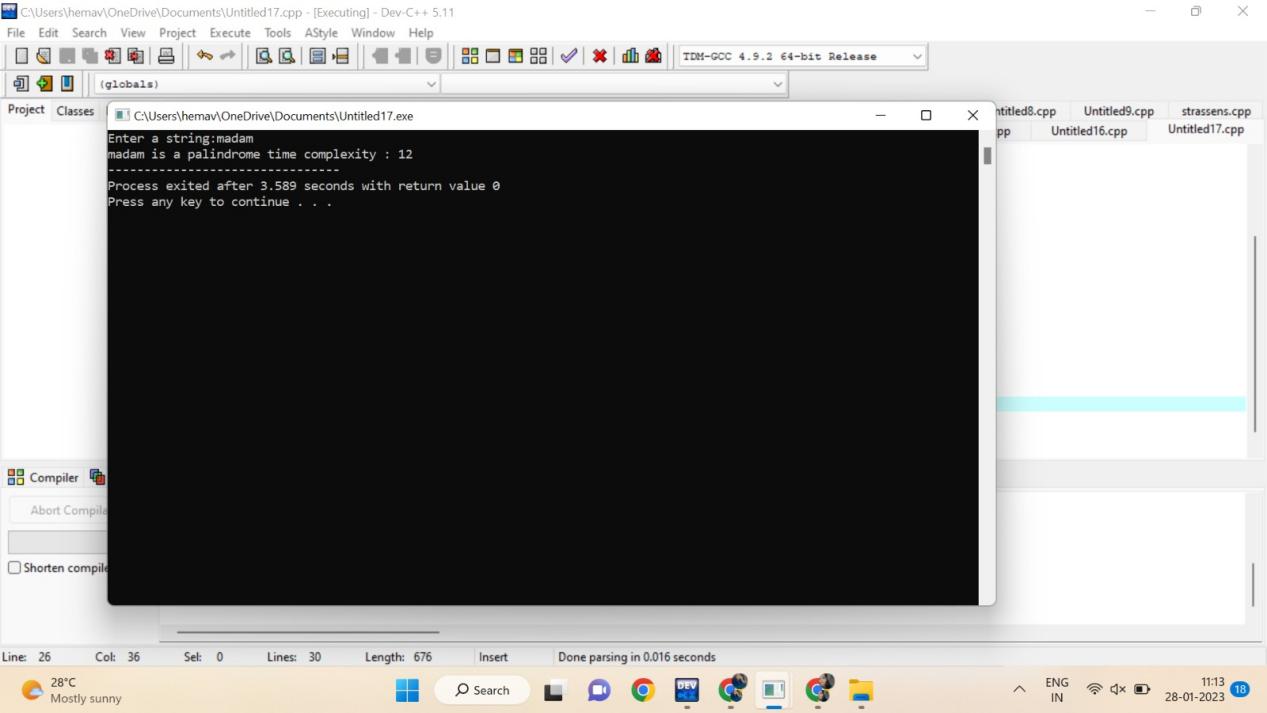
}

printf("time complexity : %d",count);

return 0;

}

OUTPUT:



13. Write a program to perform Selection sort and estimate time Complexity

Estimate the time iteration for the following set of numbers.

1. (10,5, 80,-2,5,23, 45) B. (12, 3, 0, 34, -11, 34, 22, 8) C.( 3, 35, -56, 66, 77, ,-78, 82)

CODE:

#include <stdio.h>

int main()

{

int a[100], n, i, j, position, swap;

int count=0;

printf("Enter number of elements:");

scanf("%d", &n);

printf("Enter Numbers:");

for (i = 0; i < n; i++)

scanf("%d", &a[i]);

for(i = 0; i < n - 1; i++)

{

position=i;

count++;

for(j = i + 1; j < n; j++)

{

if(a[position] > a[j])

position=j;

count++;

}

if(position != i)

{

swap=a[i];

a[i]=a[position];

a[position]=swap;

count++;

}

count++;

}

printf("\nSorted Array:");

for(i = 0; i < n; i++)

printf(" %d", a[i]);

printf(" time complexity : %d",count);

return 0;

}

OUTPUT:



14. Write a program for the given pattern the given pattern If n=4

1

1 2

1 2 3

1 2 3 4

CODE :

#include<stdio.h>

int main()

{

int rows, i, j;

int count=0;

printf("Enter the number of rows: ");

scanf("%d",&rows);

for(i = 1; i <= rows; i++)

{

for(j = rows; j > i; j--)

{

printf(" ");

count++;

}

for(j = 1; j <= i; j++)

{

printf("%d ",j);

count++;

}

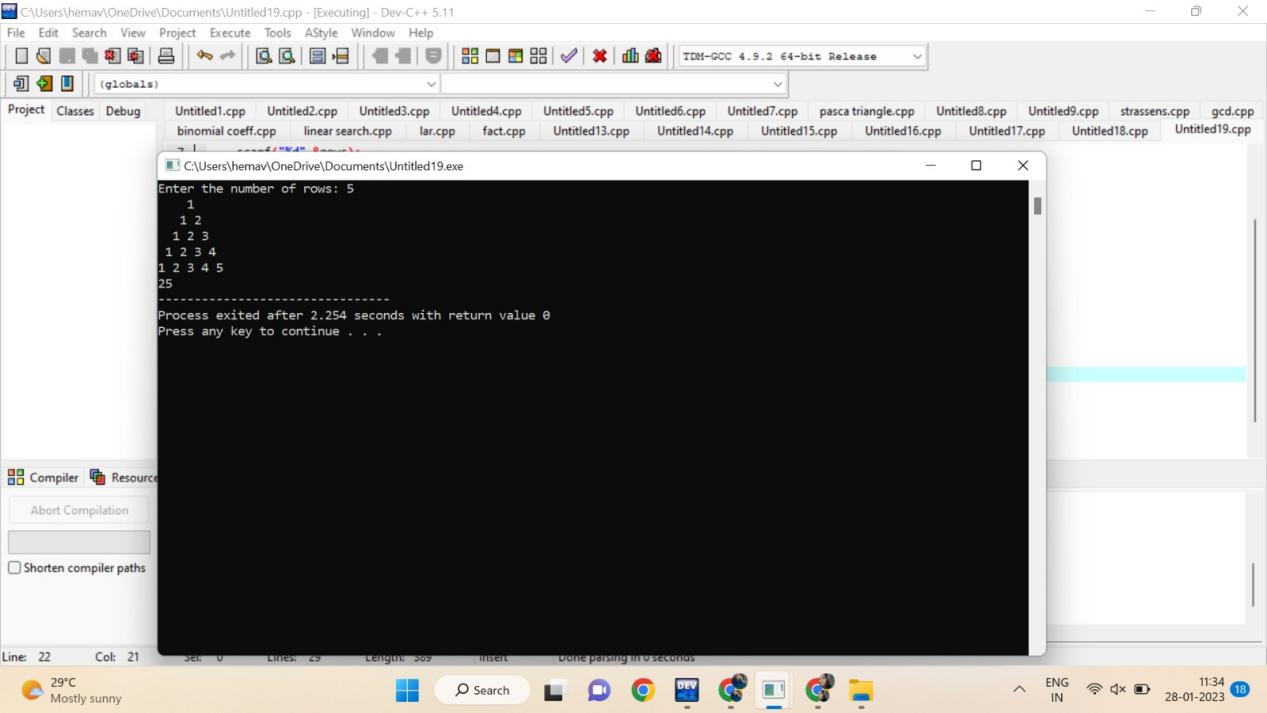
printf("\n");

}

printf("%d",count);

return 0;

OUTPUT:



15. Write a program to return all the possible subsets for a given integer array. Return the

solution in any order.

Input nums= [1,2,3]

Output : [ [], [1], [2], [3], [1,2], [1,3], [2,3], [1,2,3]]

CODE:

#include <stdio.h>

char string[50], n;

void subset(int, int, int);

int main()

{

int i, len;

printf("Enter the len of main set : ");

scanf("%d", &len);

printf("Enter the elements of main set : ");

scanf("%s", string);

n = len;

printf("The subsets are :\n");

for (i = 1;i <= n;i++)

subset(0, 0, i);

}

void subset(int start, int index, int num\_sub)

{

int i, j;

if (index - start + 1 == num\_sub)

{

if (num\_sub == 1)

{

for (i = 0;i < n;i++)

printf("%c\n", string[i]);

}

else

{

for (j = index;j < n;j++)

{

for (i = start;i < index;i++)

printf("%c", string[i]);

printf("%c\n", string[j]);

}

if (start != n - num\_sub)

subset(start + 1, start + 1, num\_sub);

}

}

else

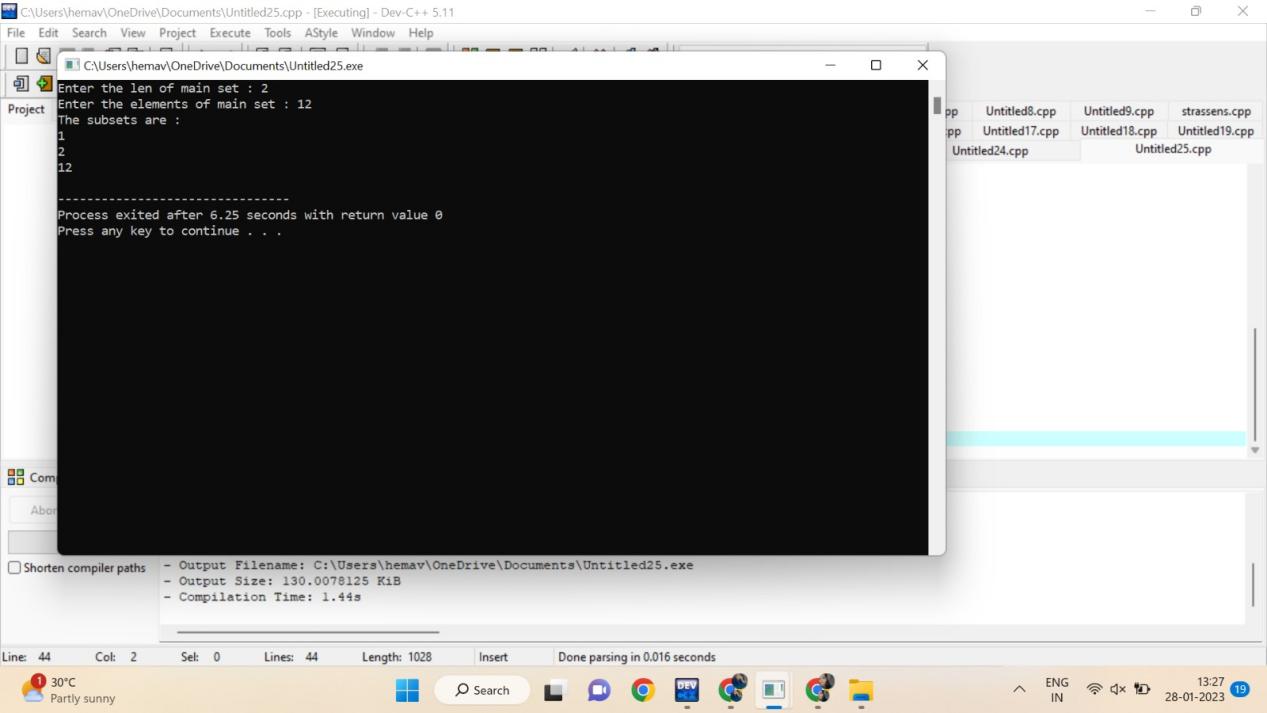
{

subset(start, index + 1, num\_sub);

}

}

OUTPUT:



16. Write a program to check sub string is there in a string or not.

Input/Output

a.orginal string = "babad" b.Orginal string = "babad" c. Orginal string = "babad"

Sub string = "shahad" Sub string = "daa" Sub string = "aba"

Output = Not Found Output = Not Found Output = Found

CODE:

#include<stdio.h>

int main()

{

char str[80], search[10];

int count1 = 0, count2 = 0, i, j, flag;

int count=0;

printf("Enter a string:");

gets(str);

printf("Enter search substring:");

gets(search);

while (str[count1] != '\0')

count1++;

while (search[count2] != '\0')

count2++;

for (i = 0; i <= count1 - count2; i++)

{

count++;

for (j = i; j < i + count2; j++)

{

count++;

flag = 1;

if (str[j] != search[j - i])

{

count++;

flag = 0;

break;

}

count++;

}

if (flag == 1)

break;

count++;

}

count++;

if (flag == 1)

printf("found");

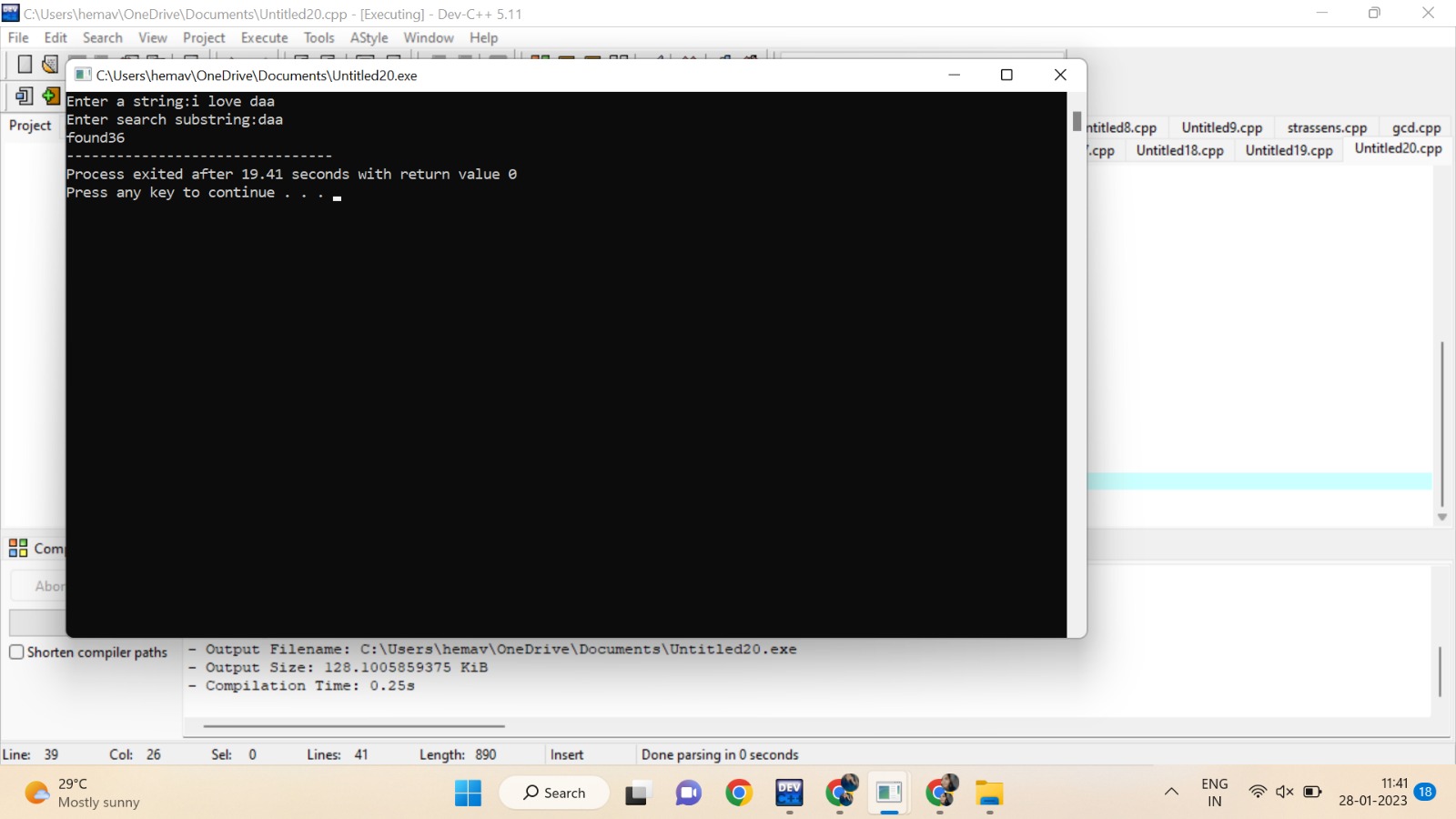
else

printf("not found");

printf("%d",count);

}

OUTPUT:



17.Write a program to perform sorting without using swapping and estimate time complexity.

CODE:

#include <stdio.h>

int main()

{

int a[100], n, i, j, position, swap;

int count=0;

printf("Enter number of elements:");

scanf("%d", &n);

printf("Enter Numbers:");

for (i = 0; i < n; i++)

scanf("%d", &a[i]);

for(i = 0; i < n - 1; i++)

{

position=i;

count++;

for(j = i + 1; j < n; j++)

{

if(a[position] > a[j])

position=j;

count++;

}

if(position != i)

{

swap=a[i];

a[i]=a[position];

a[position]=swap;

count++;

}

count++;

}

printf("\nSorted Array:");

for(i = 0; i < n; i++)

printf(" %d", a[i]);

printf(" time complexity : %d",count);

return 0;

}

OUTPUT:



18.Write a program to print the reverse of a string. And estimate the time complexity

for the given inputs.

Test cases: output –

“ as\nr5Y” Y5rn|sa

“7yut02” 20tuy7

“EryEq qEyrE

CODE:

#include<stdio.h>

Int main(){

char val[25];

printf("enter the value: ");

scanf("%s",&val);

int count=0,c=0;

while (val[count]!='\0'){

count++;

c++;

}c++;

for(int i=count-1;i>=0;i--){

c++;

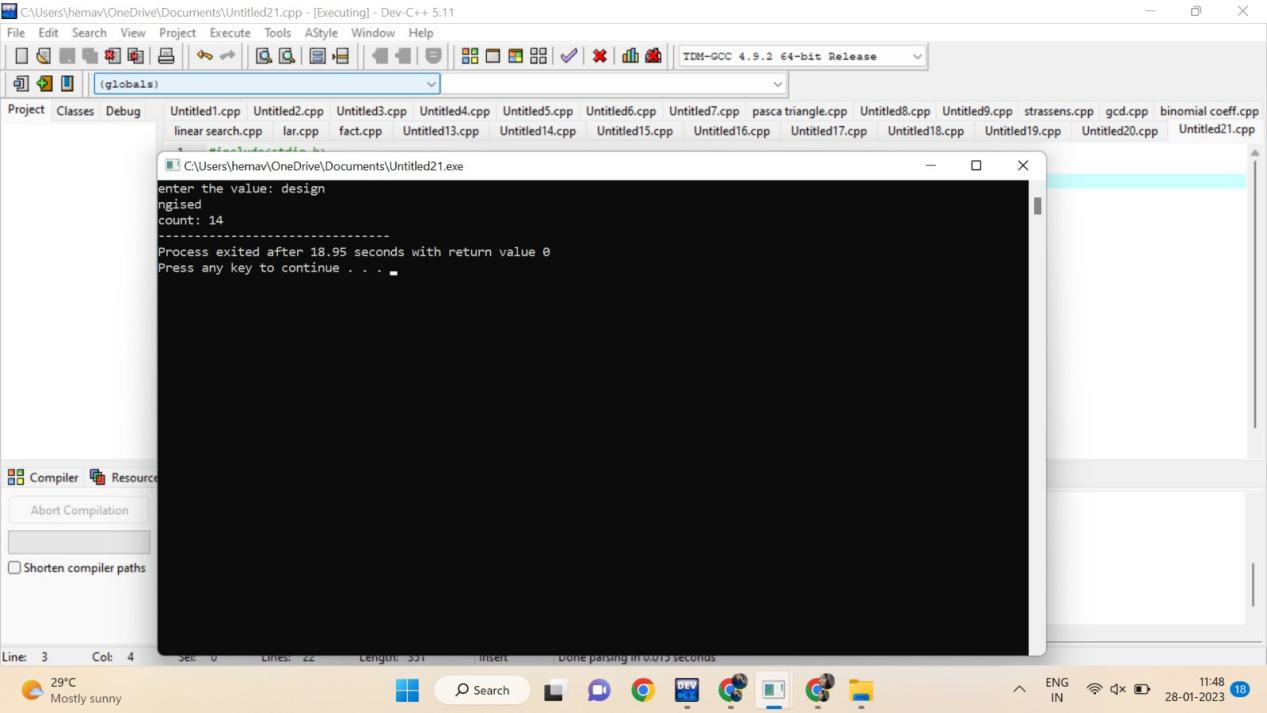
printf("%c",val[i]);

}c++;

printf("\ncount: %d",c);

}

OUTPUT:



19.Write a program to perform Bubble sort and estimate time Complexity for n values.

Perform test cases for the following set of numbers.

1. .(10,5, 80,-2,5,23, 45) B. (12, 3, 0, 34, -11, 34, 22, 8) C.( 3, 35, -56, 66, 77, ,-78, 82)

CODE:

#include<stdio.h>

void main(){

int ele,count=0;

printf("Enter tot element: ");

scanf("%d",&ele);

int arr[ele];

printf("Enter the elements: ");

for (int i = 0; i < ele; i++){

count++;

scanf("%d",&arr[i]);

}count++;

for (int i = 0; i < ele; i++)

{

count++;

for (int j =i+1; j < ele; j++)

{

count++;

if (arr[i]>arr[j])

{

count++;

int temp=arr[i];

count++;

arr[i]=arr[j];

count++;

arr[j]=temp;

count++;

}

}count++;

}count++;

printf("sorted array: ");

for (int i = 0; i < ele; i++)

{count++;

count++;

printf("%d ",arr[i]);

}count++;

printf("count: %d",count);

}

OUTPUT:

